

# Arquitetura Linux

- Kernel:
  - Suporte a drivers de diversos dispositivos
  - Gerenciamento de processamento e memória
- Shell e GUI (interface gráfica ao usuário)
- Utilitários de sistema:

Funções básicas para arquivos, como cópia, remoção, criação, etc.
- Programas

# GUIs



KDE



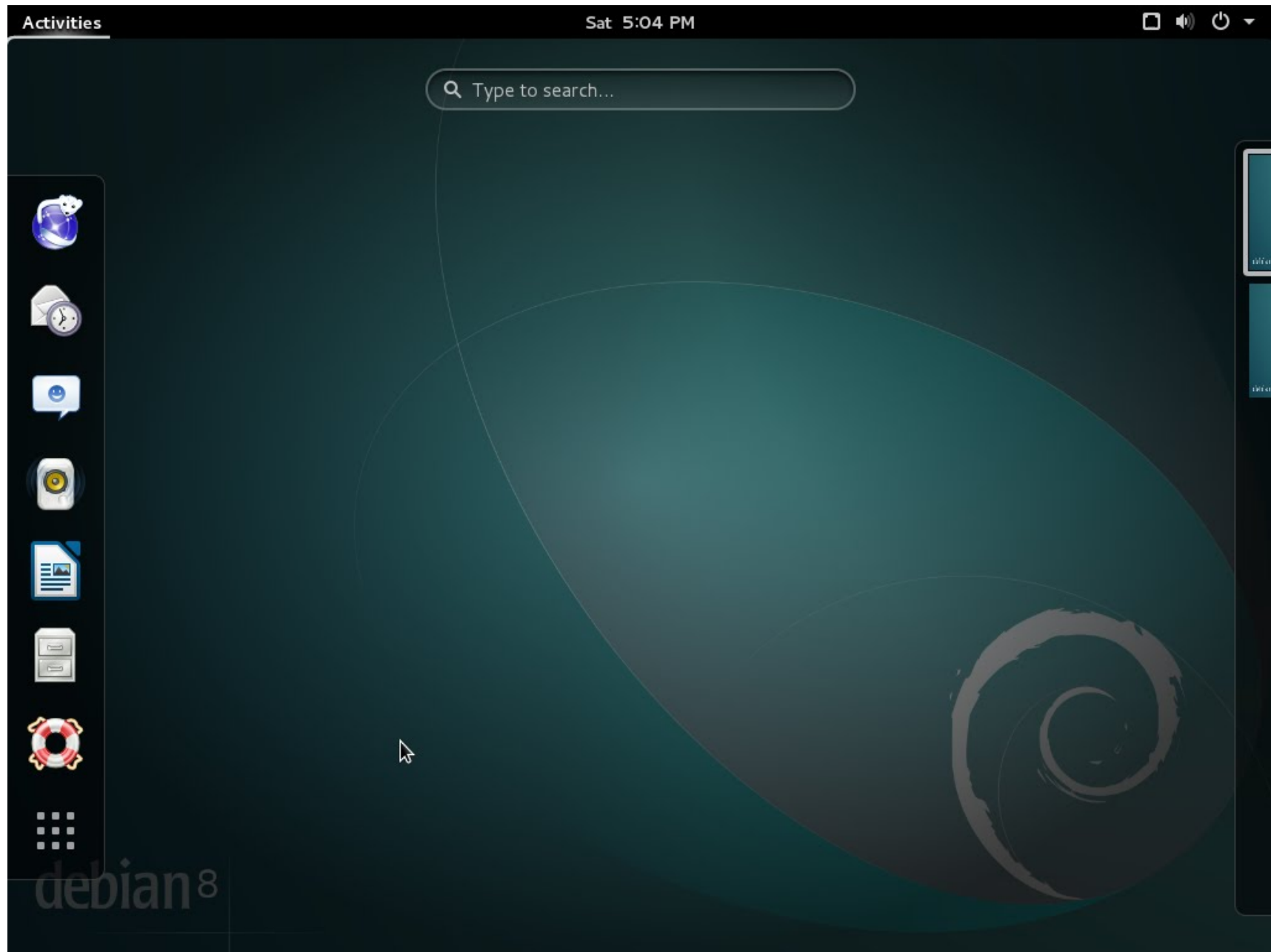
GTK+

Qt

XFCE



# GNOME = GNU Network Object Model Environment



# Gnome

- Navegador de pastas (*nautilus*) → sistemas diretórios
- TERMINAL
  - procura → terminal
  - Aplicativos → Utilitários → terminal
- adicionar terminal ao lançador
  - arrastando

ls (enter)

cd Documentos

# Sistema de diretórios

- / → diretório raiz
- /home → arquivos de usuários
- /bin → binários
- /etc → arquivos de configuração
- /var → logs e arquivos usados pelo sistema
- /lib → bibliotecas
- /dev → acesso a dispositivos físicos
- /proc → infos kernel e computador
- /home/<user> → diretório de trabalho do usuário <user>

# Introdução ao terminal

- Herança do Unix
- Permite gerenciamento básico do sistema e de arquivos e programas
- `prompt = user@máquina:pasta-atual$`  
*~ = /home/user/*

# O interpretador shell

- Atua como intermediário entre o kernel e o usuário
- É um interpretador de linhas de comando
- Funções:
  - Completa comandos [tab]
  - Guarda histórico de comandos [seta para cima]
  - Suporte a variáveis (de usuário e de sistema)
  - Suporte a scripts
  - Estruturas de comando básicas (loops, if, etc...)

# Regras da linha de comandos

- Comandos são “case-sensitive”
- O retorno da linha de comando só ocorre depois que o comando for terminado – com sucesso ou não (\*)
- O efeito de diferentes comandos pode ser afetado pelo uso de “flags”

Ex: “ls -l”

- Informações adicionais podem ser obtidas por:  
man <comando> → p/ sair “q”  
<comando> --help



# \*rodando em background (exercício)

- abrir libreoffice:

```
$ libreoffice
```

- voltar ao terminal (alt+tab ou lançador) e verificar resposta (a um enter por exemplo)

- fechar libreoffice

- abrir novamente com & no fim

```
$ libreoffice &
```

- fechar... e abrir sem o & e voltar ao terminal:

```
ctrl+z
```

```
$ bg
```

+ observações: \$ libr + tab sem & e ctrl+c ... ↑↓

# Comandos básicos terminal

`pwd`: fornece o diretório atual

`mkdir <arg>`: cria pasta <arg>

`rmdir <arg>`: remove pasta <arg>

`cd <arg>`: entra no diretório <arg>

`ls`: lista arquivos e pastas no diretório atual

`cp <arg1> <arg2>`: copia <arg1> para <arg2>\*

`mv <arg1> <arg2>`: move <arg1> para <arg2>\*

`rm <arg>`: remove <arg>

\*<arg1>=origem <arg2>=destino

Acompanham os argumentos o caminho (pasta)

# Comandos básicos terminal

```
rmdir <arg>
```

Apenas quando a pasta <arg> está vazia.

Para remover pasta e seu conteúdo recursivamente:

```
rm -r <arg>
```

# Comandos básicos terminal

## Atalhos:

- ~ = /home/user/
- .. = diretório superior
- . = diretório atual
- = diretório anterior

## Navegação por caminho relativo:

```
nicolao@trinity:~$ pwd
/home/nicolao
nicolao@trinity:~$ cd introfiscomp
nicolao@trinity:~/introfiscomp$ cd ../../
nicolao@trinity:/home$ cd -
/home/nicolao/introfiscomp
nicolao@trinity:~/introfiscomp$ cd
nicolao@trinity:~$
```

## Navegação por caminho absoluto:

```
nicolao@trinity:~$ pwd
/home/nicolao
nicolao@trinity:~$ cd /home/nicolao/introfiscomp/
nicolao@trinity:~/introfiscomp$ cd
nicolao@trinity:~$
```

# Coringas

- “\*”: substitui nenhum ou qualquer caracter
- “?”: substitui apenas um caracter
- “[a-b]”: substitui apenas um caracter, dentro dos limites “a” e “b” onde, “a” e “b” podem ser:
  - Números → ex: [0-9]
  - Letras → ex: [a-z]

ex:

```
ls /usr/bin
```

```
ls /usr/bin/gc*
```

```
ls /usr/bin/gc? - *
```

```
ls /usr/bin/gcc* [5 - 9]
```

# TAREFA 1: somente via terminal

- 1) Crie um diretório (**mkdir**) chamado "introfiscomp" no seu home
- 2) entre nesse diretório (**cd**) e crie um outro diretório chamado "nivel0", e entre nele
- 3) `$ gedit teste.txt &`  
escreva "texto nivel0" salve e saia. `$ ls`
- 4) criar pasta "nivel1" e copiar (**cp**) teste.txt para lá com nome "teste1.txt" `$ cp teste.txt nivel1/teste1.txt`
- 5) entre na pasta e modifique o texto para "... nivel1".
- 6) mover (**mv**) teste1.txt para pasta superior
- 7) voltar para pasta superior e verificar conteúdo `$ ls`
- 8) certifique-se que pasta nivel1 está vazia
- 9) remova a pasta (**rm -r**) nivel1 e remova (**rm**) o arquivo teste1.txt
- 10) renomeie (**mv**) pasta nivel0 para testeterm

# Mais alguns comandos

\$ `date` → fornece data e hora

\$ `cal` → calendário

\$ `clear` → limpa a tela (= Ctrl+L)

\$ `locate file` → mostra todos arquivos com “file” no nome

\$ `du -h pasta` → lista espaço ocupado pelos diretórios abaixo do diretório “pasta”

\$ `finger user` → exibe informações detalhadas sobre o usuário “user”

\$ `file nome` → retorna o tipo do arquivo “nome”

\$ `tar` → agrupa (concatena) ou desagrupa vários arquivos num só

```
tar -cvf textos.tar *.txt
```

```
tar -xvf textos.tar
```

```
tar -tvf textos.tar
```

# Mais alguns comandos

\$ `gzip` → **comprime** arquivos

`gzip textos.tar`

\$ `gunzip` → **descomprime** arquivos (= `gzip -d`)

`gunzip textos.tar.gz`

\$ `grep texto` → **evidencia** linhas contendo “texto”

\$ `find` → **localiza** arquivos por suas características

`find . -name “*Dirac*”`

\$ `wc arquivo` → **conta** palavras e linhas do arquivo

\$ `head arquivo` → **exibe** 1as linhas do arquivo

\$ `tail arquivo` → **exibe** ultimas linhas do arquivo

\$ `more arquivo` → **exibe** conteúdo do arquivo por págs

\$ `less arquivo` → **exibe** conteúdo + navegação (=man)

**q**=quit; **/xxx** = procura pelo padrão xxx; **b** e **n** navega nelas

\$ `cat arquivo` → **exibe** conteúdo TOTAL (concatenação)

... futuro próximo: redirecionamento entrada/saída e pipes



# Mais alguns comandos

\$ `ps` → mostra processos em execução (PID)

`ps`

`ps ux | grep "terminal"`

`ps aux | more`

\$ `kill` → envia um sinal a um processo

`kill -9 12170`

\$ `bg` → passa um processo para background

\$ `fg` → passa um processo para foreground

\$ `top` → programa de gerenciamento de processos

\$ `jobs` → lista todos processos rodando em background

\$ `nohup` → executa processo independente terminal

\$ `time` → conta tempo de execução de um programa

\$ `chmod` → altera permissões de arquivo

`chmod [ugoa] [+ -] [rwx] arquivo`

`chmod [0-6] [0-6] [0-6] arquivo`

# Editores de texto

- São diferentes de um processador de texto - word, libreoffice, etc → texto formatado
- Programas para edição de *texto simples* ou *texto puro* (ex: notepad)
  - gedit
  - nano ou pico
  - vi ou vim
  - subl
  - geany
  - brackets
  - emacs e jed
  - kate ←

# GNU Emacs

- Editor de textos extensível, altamente personalizável, interpretador e muito +

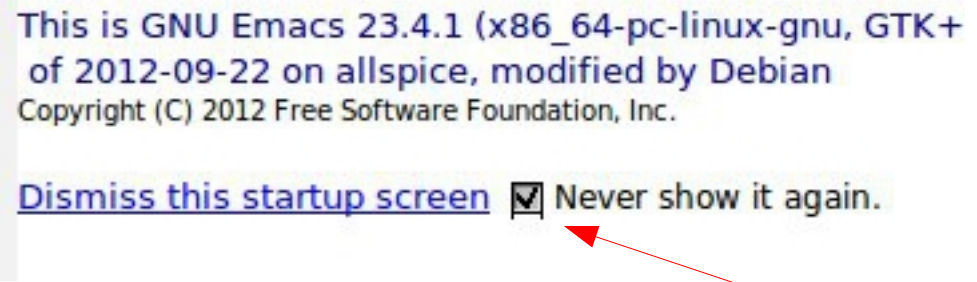
- `$ emacs`

1ª vez que abre? →

clique na caixa e selecione

“Dismiss this startup screen”

– ou dependendo da versão: `customize emacs` → inhibit startup



This is GNU Emacs 23.4.1 (x86\_64-pc-linux-gnu, GTK+ of 2012-09-22 on allspice, modified by Debian  
Copyright (C) 2012 Free Software Foundation, Inc.

[Dismiss this startup screen](#) ☒ Never show it again.

- Possui um dialeto específico de teclas de atalho:

C- = Ctrl

M- = Alt

S- = Shift

# Principais teclas de atalho emacs

- C-x C-s → salva
- C-x C-w → salvar como
- C-x C-c → sai
- C-g → aborta execução de comando parcial
- C-s → procura para frente (C-r para trás)
- C-/ ou C-\_ ou C-x u → desfaz
- C-z → minimiza
- S-setas ou C-espaco → seleciona
- C-w → recorta
- M-w ou C-Insert → copia
- C-y ou S-Insert → cola

# Outras teclas de atalho emacs

- C - ( → inicia criação de macro
- C - ) → finaliza criação de macro
- C - x C - e → executa macro
- C - x z → repete ultimo comando # vezes tecla z
- M - ! → executa comando no terminal
- C - x 1 → seleciona buffer 1 para visualização